DOI:
10.1109/ICC.2015.7249254

Document Version
Peer reviewed version

Mobility Aware Virtual Network Embedding

Giorgos Chochlidakis, *Student Member, IEEE,* Vasilis Friderikos, *Member, IEEE,*

Abstract—

Over the last years, network virtualization has become one of the most promising solutions for sustainability towards the ongoing increase of data demand in mobile networks. Within that context, the virtual network embedding problem has recently been studied extensively and many different solutions have been proposed; but mainly these studies have focused on wired networks. The main purpose of this paper is to provide an optimization framework for optimal virtual network embedding, including a heuristic algorithm with low computational complexity, by explicitly considering the effect of supporting the actual user mobility, assuming the emerging Distributed Mobility Management (DMM) scheme as well as a traditional Centralized Mobility Management (CMM) scheme. In addition to that, service differentiation is introduced, giving higher priority to time-critical over-the-top (OTT) services compared to more traditional elastic Internet applications. The performance of the proposed framework is compared to mobility agnostic greedy algorithms as well as virtual network embedding algorithms from the literature. Numerical investigations reveal that the effect of user mobility has a significant impact on the design of virtual networks. Additionally, the mobility aware scheme can provide tangible gains in the overall performance compared with the previous proposed schemes that do not take explicitly into account the effect of user mobility.

Index Terms—Virtual Network Embedding, Mobility, Network Virtualization, DMM, CMM, Network Optimization.

1 Introduction

Mobile network operators worldwide are witnessing a dramatic increase of data demand due to the introduction of smartphones and the plethora of Internet applications that they can support. At the same time, their revenues are reaching a plateau due to a flat charging model, turning them into 'dumb pipes' for the application providers [1]. For this reason, and in parallel to what has been the case in wired networks and data-centres, the virtualization of resources, in order to achieve efficient network sharing, has been considered as a promising solution for next-generation networks, including the scope of 5G. It is worth pointing out that some form of passive (non-adaptive) infrastructure sharing already exists within the cellular operators and as it has been observed, by the end of 2015, 90% of mobile operators will have explored this avenue in some form[1].

Clearly, this trend is expected to further continue in the future, but sharing will inevitably have to move deeper into the network. This will give the flexibility of more dynamic sharing via network virtualization techniques, allowing for multi-tenancy at different network elements within the core and wireless access network. The most significant advantages that network virtualization carries are the increase of utilization of the available physical resources, the energy efficiency, the flexibility and scalability that can offer and finally the prospect of sustainability for mobile operators [2] [3].

In network virtualization, the physical resources (e.g. nodes and links) are in essence virtualized into isolated slices, in order to form virtual networks, respectively to virtual network requests, in a procedure known in the literature as virtual network embedding or mapping. As already stated, the problem of virtual network embedding has been studied extensively and a plethora of different approaches have been proposed so far. A meticulous review of the research progress so far in the scope area of virtual network embedding is presented in [4] where the authors outline the most important optimization and heuristic algorithms. Then, they categorize the algorithms along three main dimensions: static versus dynamic, centralized versus distributed and concise versus redundant solutions.

An important issue that has to be taken from the outset into account while developing a virtual network embedding algorithm for mobile networks is the actual effect of users mobility. The way that the mobility support is implemented (i.e. the mobility management scheme that is deployed) affects the core network's congestion, the overall network performance and the availability of the resources. For this reason, mobility subsequently affects the virtual network embedding procedure.

Since the proposal of Mobile IPv6 (MIPv6) [5] by the Internet Engineering Task Force (IETF), various all-IP mobility management schemes have been standardized so far. The modern trend is to move from centralization to the distribution of the mobility function at the edge of the core network. To this end, after setting the requirements [6], IETF proposed the Distributed Mobility Management scheme [7], where each edge router can potentially become a mobility anchor for the users. According to DMM, when a handover occurs, the active flows are being tunnelled directly from the old to the new edge router that is connected to the new base-station. Hereafter, we will assume that mobility management is handled in a distributed manner using a DMM compliant solution, where DMM mobility anchor points are located onto the edge routers.

In this paper we consider a core mobile network physical infrastructure that has to be shared by multiple tenants after

• *G. Chochlidakis and V. Friderikos are with the Centre for Telecommunications Research, King's College London, Strand, London, WC2R 2LS, UK, e-mail: (giorgos.chochlidakis,vasilis.friderikos)@kcl.ac.uk.*

1. Mobile Network Sharing Report Developments, www.reportlinker.com

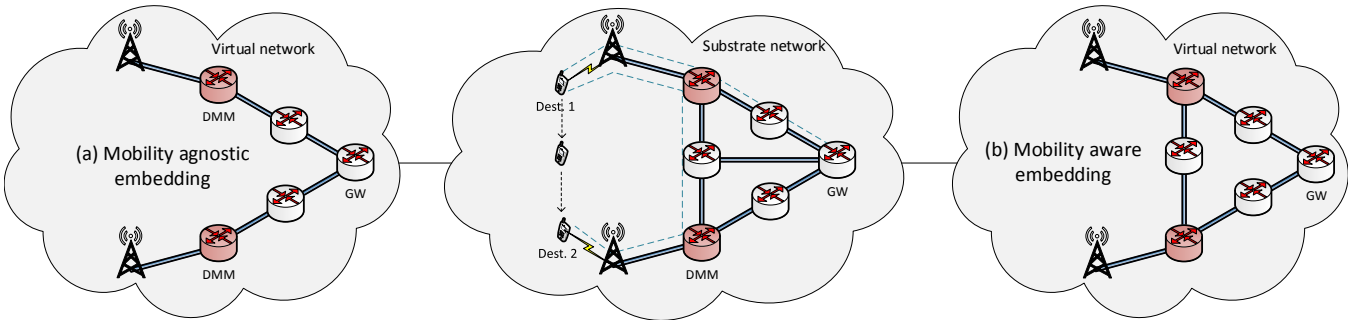

Fig. 1: (a) Mobility agnostic and (b) mobility aware embedding algorithm

different virtual networks to be efficiently set up, according to virtual network requests. The requests are classified into different classes depending on the type of the service that has to be delivered. A linear integer mathematical programming formulation is developed, which also takes into account the effect of user mobility on the design of the virtual networks. The main contribution of this work is that we reveal and examine in a conscientious manner the interaction of physical network topology and traffic tunnelling at the mobility anchors on the efficient construction of virtual networks. To this end, the proposed integer programming formulation considers in an explicit way the user mobility and the effect of that mobility on the available network resources. Also the proposed scheme performs a prioritization among multiple tenants with potential differentiation in terms of QoS but the model can be easily extended to differentiate charging business model and/or Quality of Experience (QoE).

In Fig. 1 simple example of the outcome of a mobility agnostic and mobility aware [8] virtual embedding algorithm respectively is presented. Firstly, we consider a substrate network topology, where a virtual network has to be formed in order to serve a class of flows, a fraction of which moves between the two edge routers due to expected user mobility, as depicted in the figure. The handover of the flow is handled by DMM scheme and the intermediate path between the two edge routers is deployed for the data forwarding. Then, as shown in Fig. 1a, given the gateway and the destination edge router as defined by the virtual network request, the mobility agnostic algorithm will not inevitably include in the formed virtual network the shortest routing paths that are important for the efficient handover procedure substrate tunnelling path. In this way, the DMM tunnelling will have to be routed through a path consisted of 5 intermediate physical nodes instead of the shortest one consisted of 3 nodes.

On the other hand, the proposed mobility aware algorithm takes explicitly into consideration not only the source and destination nodes but also the DMM tunnelling path for the fraction of the flows that would require mobility support. Hence, the algorithm will firstly map the two shortest paths that connect the gateway node with the base-stations (for flows that start from there) and it will also include the intermediate path as shown in Fig. 1b. In this way, the DMM tunnelling path, connecting the previous DMM edge router with the new one, will be improved, being consisted in this case, as shown in the example, of 3 nodes.

The key difference between the two approaches is that an embedding algorithm that explicitly utilizes mobility information will map the required paths that will be used for the data tunnelling to the next mobility anchor point where the flow will migrate. Considering the users mobility effect, such a mobility aware algorithm will create virtual networks that can support migrating flows in a significant more efficient manner. To the best of our knowledge, this is the first time a mobility aware optimal virtual network embedding algorithm is developed.

The remainder of this paper is organized as follows: in section 2 a brief review of the concept of Software Defined Networking (SDN), Network Function Virtualization (NFV) and the main research approaches on virtual network embedding algorithms from the literature are given. In section 3 the proposed algorithm and the algorithms that we compare to it are presented. In section 4 the simulation results are explained and commented and finally in section 5 the concluding remarks from this work are presented.

2 Network Virtualization via SDN, NFV & Virtual Network Embedding

2.1 Software Defined Networking

The need for scalable, dynamic and resource efficient next generation networks has made Software Defined Networking (SDN) [9] the dominant architecture that explicitly can exploit the benefits that network virtualization offers. SDN decouples the control plane from the data forwarding plane and abstracts the network function. In this way, it offers the freedom to dynamically configure the forwarding logic.

The network intelligence is logically centralized in a software-based SDN controller (or more, depending on the scale of the network). The SDN controller has full knowledge about the state of the physical infrastructure and its responsibility is to updated timely network policies according to flow activities. Moreover, application programming interfaces (APIs) are supported to link the application and the control layer in order to facilitate the network management (Fig. 2). Acceleration on developments on the SDN frontier is expected to increase as various collective efforts mature such as the Open Networking Foundation ONF [2], the OpenDaylight [3] and the OpenCompute [4].

2. www.opennetworking.org
3. www.opendaylight.org
4. www.opencompute.org

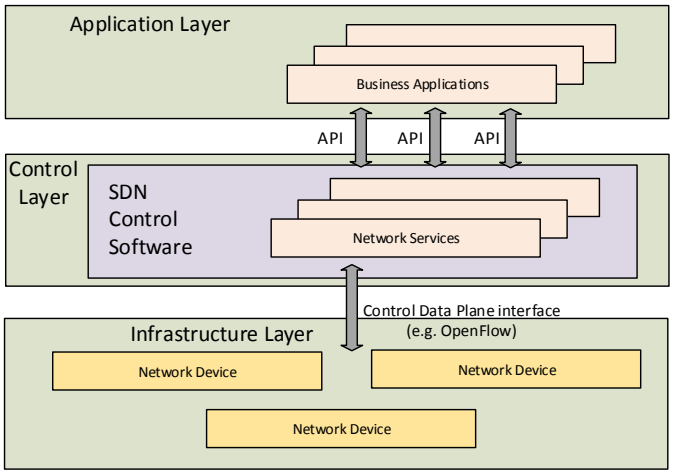

Fig. 2: SDN architecture overview

Regarding the enabling of SDN, OpenFlow [10] protocol can be used as a standard communications interface between the control layer and the forwarding layers. In particular, it provides access to the forwarding plane of physical and virtual network devices (i.e. switches, routers). The programmable OpenFlow routers and switches follow the policies that are determined by the SDN/OpenFlow controller. Within this remit some other protocols could be utilized such as for example *NETCONF*, which provides mechanisms for installing, manipulating and deleting set of defined configurations at SDN enabled network devices [11].

Flowvisor [12] is an innovative network virtualization approach that acts like a hypervisor between software and hardware on a PC using OpenFlow as a hardware abstraction layer to sit logically between control and forwarding paths on a network device. The same researchers who proposed FlowVisor in [13] describe a way to build a testbed that is embedded in the network and evaluates its performance. The main advantages of using FlowVisor in a sharing infrastructure (like the one in the scenario that we will investigate) is that it provides transparency, strong isolation and dynamic slice definition.

Another interesting SDN approach that can also be implemented in the hereafter presented scenario, is given in [14]. The authors believe that slice isolation should be provided at the language level and to this end, they propose a slice abstraction that facilitates the isolation between network programs. They firstly define a simple programming abstraction for defining slices and then they describe algorithms for compiling slices to OpenFlow switches. Then they present the performance of their proposal. For a more detailed treatment on current and emerging SDN approaches, a notable work which surveys the state-of-the-art in traffic engineering for SDNs is given in [15].

2.2 Network Function Virtualization

Network Function Virtualization (NFV) [16] proposes the installation of the mobile network functions as software instances on data-centres, network nodes and on the end users' premises. This aims to limit the hardware dependence, decrease the maintenance costs, energy [17] and capital investment and increase the network scalability and robustness. NFV is a highly promising solution for any data plane packet processing and control plane function in wired or wireless mobile network infrastructures.

A combination of both SDN and NFV implementation can be seen in [18] where the authors argue that depending on network performance in terms of load and delay, there can be areas of the network topology where NFV deployment might be more beneficial and other areas where SDN deployment with decoupled data and control plane performs better. Then a function placement problem is proposed that minimizes the transport network load overhead with respect to different parameters.

NFV bears the potential of flexibly locating network functionalities in a vendor agnostic networking hardware and also there is the potential of combining network functions in on-demand adaptive way, hence creating a link with SDN. Although NFV bears plenty of advantages and may be a step in the right direction, it might be insufficient [19]. The authors believe that it is very important that a programmable and dynamic realization of per-flow control that manages flows across different functions chains should be adopted for next-generation networks. Such an approach can allow service providers to quickly create and deploy new revenue-generating services.

2.3 Related virtual network embedding algorithms

Different approaches have been proposed so far addressing the virtual network embedding problem. In this paper we include selected related works from the literature. Firstly, the authors in [20] formulate the virtual embedding problem as a mixed integer program and then propose virtual embedding algorithms by introducing a coordination between node and link mapping phases. More specifically, they relax the integer constraints to get a linear programming setting by using deterministic and randomized rounding techniques and so they manage to achieve polynomial-time albeit suboptimal solutions for virtual network embedding. The simulation results show that their algorithms outperforms the existing approaches in terms of acceptance ratio, revenue and provisioning cost.

The authors in [21] propose an linear integer programming formulation that solves the online virtual network embedding problem. Their solution aims to minimize the total resource consumption and to achieve load balancing at the network. To this end, they introduce three cost functions: one that minimizes the total load on each virtual network, one that minimizes the number of links that are mapped and selects nodes with higher available resources and one that includes the demanded capacity by the virtual network requests in the objective function. The simulation results show that their proposal outperforms in general the different compared heuristic approaches.

A virtual network embedding distributed protocol, named *MADE* suitable for mobile environments, where the nodes are not static and the substrate network is dynamic is presented in [22]. The authors apply the path splitting and migration techniques in order to optimize the utilization of the available physical resources. The simulation results show the efficiency of this protocol in terms of acceptance and completion of the requested virtual networks.

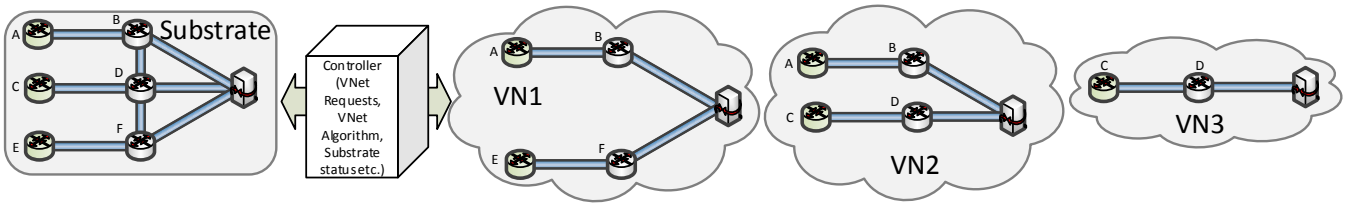

Fig. 3: Forming virtual networks on the top of a core network infrastructure under a predefined strategy

In [23] a distributed algorithm, which performs load balancing and virtual network embedding over a substrate network, is presented. The algorithm makes use of a proposed mapping protocol that enables the communication and the exchange of messages between the substrate network's nodes in order to ensure distributed negotiation and synchronization for the virtual network set up. The performance results show that in the distributed mapping approach the number of messages exchanged can have an important impact on the overall performance but, on the other hand, compared to the centralized approach, their proposed algorithm can reduce the time delay and process multiple parallel virtual network requests.

The authors in [24] develop a scalable embedding algorithm named *VNE-AC*, based on ant colony metaheuristic. The algorithm aims to minimize the allocated physical resources of the physical substrate network for each request in order to minimize the reject rate and to maximize the provider's revenue. Results from the simulations show that the proposed algorithm achieves better overall performance in comparison to related algorithms from the literature.

In [25] is presented an alternative approach for the virtual embedding problem, which focuses on rethinking the designing of the substrate network per se in order to enable less complex algorithms and increase the utilization of the resources without the restriction of the problem space. In order to achieve flexibility of the substrate network, path splitting and migration as well as customized node-mapping algorithms are used. The simulations results show that the proposed solution is competitive and it manages to increase the resources' utilization.

In [26], the authors focus on solving the problem of virtual network embedding as the substrate network evolves. To this end, they present an integer programming formulation of this problem which aims to minimize the upgrading cost of virtual network with respect to node resource and path delay constraints. Because of the problem's complexity the authors develop a heuristic algorithm and they present its efficiency through simulations.

A different approach is presented in [27] where the main purpose of the proposed virtual embedding algorithm is to deploy virtual networks according to the client's requirements in a reliability context with bandwidth feasibility and reliability calculations.

Finally, the authors of [28] focus on two categories of the virtual network assignment problem: virtual network assignment without configuration and with configuration. They try to provide heuristic algorithms and optimization strategies in order to increase the efficiency in the resources utilization and to be able to deal with the demands real-time. One of the proposed algorithms will be presented extensively in the next section and will be used in order to compare our proposal.

In all the above previous research works the effect of mobility has not been considered and, as shown in the sequel, this is an important parameter that needs to be taken into account for creating more efficient virtual networks.

3 System Model

In order to evaluate the performance of our proposed algorithm, we compare it to different approaches under the same assumptions. Firstly a substrate core network is mathematically formulated. The aim is to form virtual networks under requests by combining the physical resources, under specific constraints. Then we evaluate the performance of the virtual networks under the same traffic model and mobility of the users.

3.1 Implementation of virtual network embedding algorithms

We consider that in order the virtual network requests to be served, a central controller, as discussed in section 2.1, exists, being responsible for slicing the physical resources and providing isolation among them. In particular, the controller is handled by one of the hereafter algorithms (Fig. 3). This means that the embedding algorithm creates a plan/strategy for virtualization of the resources and for their assignment to the different requests. Each algorithm has different strategy and logic regarding the most efficient mapping, depending on the parameters that it takes into consideration.

After the decision for the assignment of the resources, the controller is responsible for communicating with the network elements and forming the slices. In this way, the virtual networks are created and they are ready to serve the flows which they are responsible to handle. Each slice is totally isolated from the others and has no awareness of the existence of the rest slices.

Regarding the architecture, for our topology, we consider a single gateway and several destination edge routers. Each virtual network request is defined by demands from the source gateway to the end edge routers. Hence, the formed virtual networks have a substrate topology dependence and should include the same source and destination nodes.

As for the mobility management function, since DMM scheme is deployed, every single destination edge router acts as a mobility management anchor point. Hence, we consider that by applying SDN and NFV deployment, we virtualize the mobility function by co-locating it at the edge of the core network (i.e. the edge routers).

3.2 Optimal mobility aware virtual network embedding

In this section we present the proposed mathematical programming setting for optimal, in terms of routing cost, mobility aware virtual embedding. In order to provide a formal model for the virtual network embedding process and in order to develop the optimization framework, firstly we detail the physical substrate network, then the requests for virtual networks, the objective function and the problem constraints.

3.2.1 Substrate network

The substrate network can be modelled as an undirected planar tree-like graph $G = (V, E)$, where the set V represents the set of nodes and the set E depicts the set of links. Let $B : E \rightarrow \mathbb{R}_{>0}$ be the cost of each link and $C : V \rightarrow \mathbb{R}_{>0}$ the capacity of each node. The gateway is located at the root of the graph, while the leaves of the tree represent the destination edge routers. The intermediate nodes are the routers of the core network. The algorithm objective is to assign nodes and links according to the virtual network requests in a way that it will minimizes the total weighted routing cost.

3.2.2 Virtual network requests

The proposed algorithm offers a multi-tenant prioritization. This can be applied in a case where the infrastructure owner provides a tenant differentiation according to desirable QoS requirements. This, also, applies to a scenario where a prioritization in favour of one type of service over another one with lower priority is required. For this reason we will refer to this feature of our proposed algorithm as tenant or type of service classification/differentiation.

Let Q represent the set of sets of the virtual network requests for all tenants. Moreover, let U represent the set of the different tenants. Then, let $Q^u \in U$ represent the set of virtual network requests for a tenant $u \in U$. The set Q can be described as follows: $Q = Q^{u_1} \bigcup Q^{u_2} \bigcup ... \bigcup Q^{u_{|U|}}$.

In order to define the set of virtual network requests $Q^u \in Q$ we consider that each request $q \in Q^u$ is described by a set of classes of unsplittable flows $k \in K^{uq}$ that have to be satisfied and routed from the root gateway to the edge routers $j \in T$ by the formed virtual networks. Also, each class of flows $k \in K^{uq}$ corresponds to a total demand d_k^{uq}.

Hence the virtual network request $q \in Q^u$ can be represented as a graph $G' = (V', E')$ that has to be mapped on the graph $G = (V, E)$. The graph $G' = (V', E')$ is consisted of virtual paths $\pi_{kp} \in P$ that connect the virtual gateway with the virtual edge routers. Each class of flow of a virtual network request can be considered as a demand for resource allocation across the core network.

We hereafter explicitly consider a per-class allocation of the available network resources allowing for scalability and assuming that requested traffic demand incorporate sufficient slack values, so that traffic variations (congestion episodes) during the duration of the virtual network are taken into account.

In order to capture the actual mobility of the users, we consider a handover matrix $H_{K\times K}$, the elements $h_{kj} \in (0, 1)$ of which represent the probability of the flow $k \in K^{uq}$ to migrate to another edge router $j \in T$ (note that each edge router j can be described by a flow $k \in K^{uq}$). When DMM scheme is in use, there is an additional need to assign paths that connect edge routers or network elements where DMM anchoring is taking place.

In this work and without loss of generality we assume that DMM anchoring is taking place at the edge router and for this reason, for each edge router of class of flow $k \in K^{uq}$ towards an edge router $j \in T$, we consider the set of substrate paths $r_{kji} \in R$. In the same way, each tunnelled class of flows will be routed through one of the available paths.

However, our optimization algorithm can be adapted for different other mobility management solutions, since mobility cost depends on the mobility management scheme that is used. In conclusion, the algorithm, taking into account the expected probabilities of handovers, will also map paths for the virtual networks for more efficient data forwarding.

3.2.3 Problem variables

Based on the above setting, the goal is to find the optimal selection of routing paths in order to minimize the total routing cost and at the same time to achieve tenant differentiation. Below, there is a summary of the variables that used for the formulation of the integer mathematical program and the introduction of the decision variables:

- $G = (V, E)$: undirected planar graph
- π_{kp}: set of substrate paths from source to edge router k (expressed in routing cost)
- r_{kji}: set of substrate paths from edge router defined by flow $k \in K^{uq}$ to edge router $j \in T$ (expressed in routing cost)
- U: set of tenants
- Q^u: set of virtual network requests of tenant $u \in U$
- K^{uq}: set of classes of flows that have to be served by virtual network of request $q \in Q^u$
- d_k^{uq}: set of demands for class of flows $k \in K^{uq}$ of tenant $u \in U$ and virtual network request $q \in Q^u$
- $z_{kp}^n = \begin{cases} 1, \text{ if node } n \in \pi_{kp} \\ 0, \text{ otherwise} \end{cases}$
- $l_{kji}^n = \begin{cases} 1, \text{ if node } n \in r_{kji} \\ 0, \text{ otherwise.} \end{cases}$

Then, we define the following problem binary variables:

$$x_{kp}^{uq} = \begin{cases} 1, \text{if } \pi_{kp} \text{ is assigned to k } \in K^{uq} \\ 0, \text{ otherwise} \end{cases} \quad (1)$$

$$y_{kji}^{uq} = \begin{cases} 1, \text{if } r_{kji} \text{ is assigned to k } \in K^{uq} \\ 0, \text{ otherwise.} \end{cases} \quad (2)$$

3.2.4 Objective function and problem's constraints

The total routing cost ϕ can be written as:

$$\phi = \sum_{u\in U} \sum_{q\in Q^u} \sum_{k\in K^{uq}} \sum_{p\in P} d_k^{uq} \pi_{kp} x_{kp}^{uq} \quad (3)$$

and expresses the cumulative routing cost of the aggregated set of flows that will use the formed virtual networks with the expected demands. The total mobility cost M can be written as:

$$M = \sum_{u\in U} \sum_{q\in Q^u} \sum_{k\in K^{uq}} \sum_{j\in T} \sum_{i\in I} h_{kj} d_k^{uq} r_{kji} y_{kji}^{uq} \quad (4)$$

and it is the routing cost of the forwarded traffic caused by the mobility of the users.
The total cost T is the summation of the routing cost and the mobility cost:

$$T = \phi + M \tag{5}$$

and the total cost T_u for the tenant $u \in U$:

$$\sum_{q \in Q^u} \sum_{k \in K^{uq}} \Big(\sum_{p \in P} d_k^{u_1 q} \pi_{kp} x_{kp}^{u_1 q} + \sum_{j \in T} \sum_{i \in I} h_{kj} d_k^{u_1 q} r_{kji} y_{kji}^{u_1 q} \Big) \tag{6}$$

Based on the above definitions the mathematical program can be formulated as follows:

$$\textbf{minimize } T \tag{7}$$

subject to

$$\sum_{u \in U} \sum_{q \in Q^u} \sum_{k \in K^{uq}} \Big(\sum_{p \in P} d_k^{uq} x_{kp}^{uq} z_{kp}^n + \sum_{j \in J} \sum_{i \in I} h_{kj} d_k^{uq} y_{kji}^{uq} l_{kji}^n \Big) \leq C_n \text{ , } \forall n \in V \tag{8}$$

$$w_n T_{u_n} \leq w_{n+1} T_{u_{n+1}} \text{ , } \forall u, q, k, p, i \tag{9}$$

$$\sum_{j \in J} \sum_{i \in I} y_{kji}^{uq} = \mathbb{I}(h_{kj}) \text{ , } \forall \ u, q, k \tag{10}$$

$$\sum_{p \in P} x_{kp}^{uq} = 1 \text{ , } \forall \ u, q, k \tag{11}$$

$$x_{kp}^{uq}, y_{kji}^{uq} \in \{0, 1\}, \ \forall \ u, q, k, p, i \tag{12}$$

where constraint (8) makes sure that the capacity of each node is not violated. Constraint (9) ensures the classification of the two types of service by introducing the weighted constraint. Moreover, constraint (10) guarantees that if there is a handover between two ARs (defined by matrix h, then one and only one path will be used. Note that $\mathbb{I}$ is an indicator function defined as follows:

$$\mathbb{I}(n) = \begin{cases} 1, \text{ if } n \neq 0 \\ 0, \text{ if } n = 0. \end{cases} \tag{13}$$

Finally, constraint (11) ensures that among all the alternative paths which connect the source node with an edge router k only one will be used and constraint (12) makes sure that the decision variables will be boolean. Since integer linear programming is NP-complete [29], problem instances related to large networks might be intractable and so low complexity heuristic methods are useful for such instances. In the next subsections, heuristic approaches from the literature and proposed by us are presented in order to evaluate the performance of the proposed scheme.

3.3 Greedy mobility agnostic heuristic algorithm

For the first heuristic approach of the above problem, we implement a generalized greedy algorithm as the heuristic algorithm presented in [25] (*Algorithm 1 - Greedy Node Mapping Algorithm*), by augmenting it to multiple traffic classes.

The algorithm firstly sorts the demands of every virtual network request of each tenant. Then, it handles firstly the high-priority one in the following way: the flows with the higher aggregate demands utilize the lower-cost routing paths, until a threshold on the congestion level of these paths is reached, and then the flows with the lower aggregate demands follow. After the high-priority tenant has been assigned with routing paths which form the requested virtual network, the algorithm serves the second tenant, which has lower priority.

The steps of this algorithm are presented in Alg. 1. Note that the mobility agnostic greedy algorithm embeds networks where the mobility is not taken into account. In this way, the tunnelling of the flows from the previous to the new DMM edge routers will use the mapped paths, which connect the gateway with the edge nodes (as we already explained at the exampled shown in Fig. 1).

Algorithm 1: GREEDY MOBILITY AGNOSTIC HEURISTIC ALGORITHM

INPUTS: Graph $G = (V, E)$, number of tenants U, virtual network requests Q, classes of flows K, set of paths P, set of demands d, set of capacities C
OUTPUTS: Set of mapped substrate paths for every virtual network request

```
for u = 1 to U do
  sort (descending) demands d^{uq} ∀q ∈ Q, k ∈ K
  for q = 1 to Q do
    for k = 1 to K do
      p = 1
      repeat
        if path π_{kp} not congested then
          map path π_{kp} to request q
          decrease available capacity of path π_{kp}
          flag = 1
        end if
        p = p + 1
      until flag = 1 or p > P
      flag = 0
    end for
  end for
end for
```

3.4 Greedy mobility aware heuristic algorithm

After implementing the greedy node mapping algorithm as it was proposed in [28] for our case, we extend it in order to be able to take into account the mobility effect.

In particular, the algorithm is like Alg. 1 but also assigns intermediate paths for the data forwarding between the edge routers, during the handover procedure. Hence, the algorithm, also, sorts the available pre-calculated paths that connect the edge routers and it assigns them to the virtual network requests, supporting firstly the high demanded ones. In addition, like the previous two presented algorithms, this algorithm also performs tenant prioritization. The steps of this algorithm are summarized in Alg. 2.

3.5 Basic virtual network assignment mobility agnostic algorithm

After having implemented two greedy heuristic approaches we are interested in implementing another virtual network algorithm from the literature which takes into account the

Algorithm 2: GREEDY MOBILITY AWARE HEURISTIC ALGORITHM

INPUTS: Graph $G = (V, E)$, types of services U, virtual network requests Q, classes of flows K, set of paths P and R, set of demands d, set of node capacities C and handover matrix $H_K \times K$
OUTPUTS: Set of mapped substrate paths for every virtual network request

for $u = 1$ to U **do**
  sort (descending) demands $d^{uq}\ \forall q \in Q, k \in K$
  for $q = 1$ to Q **do**
    for $k = 1$ to K **do**
      $p = 1$
      repeat
        if path π_{kp} not congested **then**
          map path π_{kp} to request q
          decrease available capacity of path π_{kp}
          $flag_1 = 1$
          for $j = 1$ to K **do**
            $i = 1$
            repeat
              if path r_{kji} not congested **and** $h_{kj} \neq 0$ **then**
                map path r_{kji} to request q
                decrease available capacity of r_{kji}
                $flag_2 = 1$
              end if
              $i = i + 1$
            until $flag_2 = 1$ **or** $i > R$
            $flag_2 = 0$
          end for
        end if
        $p = p + 1$
      until $flag_1 = 1$ **or** $p > P$
      $flag_1 = 0$
    end for
  end for
end for

node and link stresses. In particular, we are interested in comparing our proposal to *Algorithm 1 - Basic VN Assignment Algorithm* that is presented in [28].

In order to adapt this algorithm for our case, we firstly assume that the algorithm maps the virtual source (gateway) to the physical one and the virtual sink nodes to the physical edge routers.

Then, it maps paths which have the minimum Π upon the i^{th} arrival of a virtual network request, as it is defined below:

$$\Lambda(\pi_{kp}) = \frac{\sum_{e \in p_{kp}} \frac{1}{S_{lmax}(i) + \delta_L - S_L(i,e)}}{S_{nmax(i)+\delta_N - S_N(kp,i)}}, \tag{14}$$

where e denotes a link of a path p_{kp}, S_{lmax} and S_{nmax} the current maximum link and node stress respectively and $\delta_L = \delta_N = 1$ a relatively small number to avoid dividing by zero.

Regarding the notion of stress, for a substrate node n the stress $S_N(n)$ is defined as the number of the virtual nodes that are assigned to it. So, the stress $S_N(kp, i)$ of a path p_{kp} is the summation of the stresses of all of the nodes that belong to this path. For a substrate link e the stress $S_L(e)$ is the number of virtual links that have been mapped to it.

This algorithm aims to perform a load balancing across the network by attempting to minimize the stress of the nodes and the links. We also consider a *one-to-one* mapping; a virtual network element is mapped to a single substrate network element. The exact steps of this algorithm can be seen in Alg. 3.

Algorithm 3: BASIC VN ASSIGNMENT MOBILITY AGNOSTIC ALGORITHM

INPUTS: Graph $G = (V, E)$, number of tenants U, virtual network requests Q, classes of flows K, set of paths P, set of demands d, set of capacities C
OUTPUTS: Set of mapped substrate paths for every virtual network request

for $u = 1$ to U **do**
  for $q = 1$ to Q **do**
    for $k = 1$ to K **do**
      $p = 2$
      repeat
        if path π_{kp} not congested **and** $\Lambda(\pi_{kp}) < \Lambda(\pi_{k(p-1)})$ **then**
          map path π_{kp} to request q
          decrease available capacity of path π_{kp}
          $flag = 1$
        end if
        $p = p + 1$
      until $flag = 1$ **or** $p > P$
      $flag = 0$
    end for
  end for
end for

3.6 Randomized mobility agnostic heuristic algorithm

Lastly, we implement a virtual network embedding algorithm which assigns paths in a completely randomized way. This means that for every request, regardless the class U, the algorithm forms the network by choosing randomly routing paths among the set of the available paths. This approach does not perform any optimization strategy and it does not take the users mobility effect into account. The detailed steps can be seen in Alg. 4. In addition it doesn't perform any kind of tenants classification.

We have to note that if the capacity of the substrate network is much higher than the total demand of the flows, then this algorithm, because of the fact that it uses no intelligence, is much faster to be solved and there is a high probability of avoiding the rejection of the virtual network requests.

4 PERFORMANCE EVALUATION

In this section we evaluate the performance of the above presented virtual network embedding algorithms. Our in-

Algorithm 4: RANDOMIZED MOBILITY AGNOSTIC HEURISTIC ALGORITHM

INPUTS: Graph $G = (V, E)$, number of tenants U, virtual network requests Q, classes of flows K, set of paths P, set of demands d, set of capacities C
OUTPUTS: Set of mapped substrate paths for every virtual network request

```
for u = 1 to U do
  for q = 1 to Q do
    for k = 1 to K do
      repeat
        p = randi(1 : P)
        if path π_kp not congested then
          map path π_kp to request q
          decrease available capacity of path π_kp
          flag = 1
        end if
      until flag = 1
      flag = 0
    end for
  end for
end for
```

TABLE 1: Simulation Parameters

Parameter	Value
Network nodes (V)	31
Types of services (U)	2
Requests per service (Q)	2
Flows per request (K)	16
Alternative paths (P)	5
Alternative paths (R)	5
Flow demand (d)	1
Node capacity ((C)	100
Probability of one-hop handover (s)	70%
Probability of two-hop handover (w)	40%

tention is to observe the effect of aggregated end-users mobility and how each one of the presented algorithms performs. Then, we compare the performance of our virtual network embedding mobility aware algorithm to the rest of the previously presented algorithms.

4.1 Distributed Mobility Management scheme

At the first part of the simulations we are interested in evaluating the performance of the virtual network embedding algorithms in cases where the available physical resources are sufficient to afford all of the requested traffic.

The parameters of the simulation scenario can be seen in Table 1. In order to solve the proposed integer programming optimization algorithm we used MATLAB's optimization toolbox. The rest of the algorithms were simulated using MATLAB as well.

4.1.1 Total routing cost T

In Fig. 4 the overall performance of the aforementioned algorithms as the total ratio of migrating traffic due to mobility to the total traffic increases can be seen. In particular, there is a comparison of the total cost as it was calculated in the objective function (6) of the optimization framework.

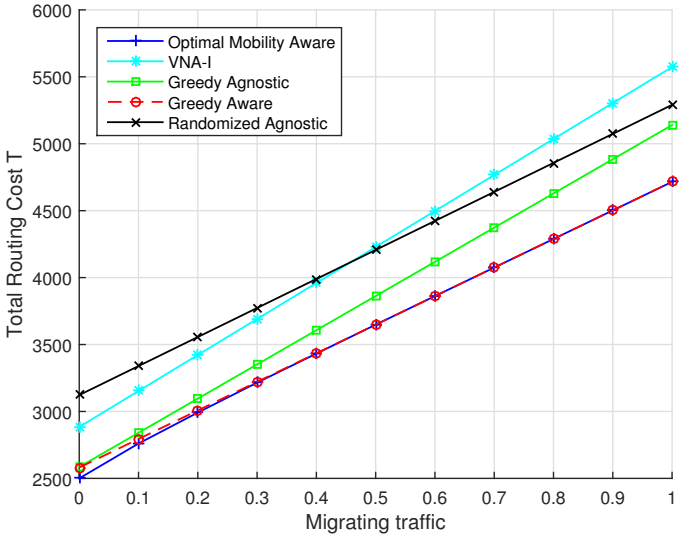

Fig. 4: Total cost as the mobility increases

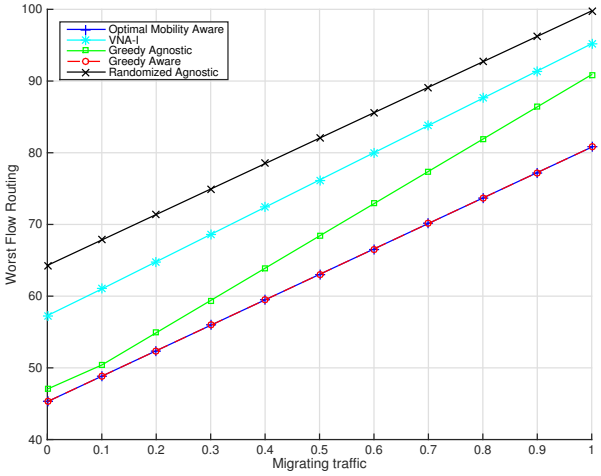

Fig. 5: Worst flow cost as the mobility increases

The two mobility aware algorithms outperform the rest of the algorithms in this low congestion scenario, achieving the same total cost. Note that the greedy mobility agnostic algorithm has significantly lower performance, up to 10% and the VNA-I along with the randomized agnostic algorithm achieve the worst total cost. As it was expected, since the network is not heavy loaded, the greedy algorithm manages to embed firstly the best paths and for this reason it has the same performance with the optimization algorithm that provides optimal solutions.

In Fig. 5 there is a presentation of the worst routing cost of all the flows against the percentage of the mobility for each one of the algorithms. A linear behaviour can be observed as with respect to the total cost. In this scenario as well, the mobility aware algorithms have the least worst case performance whereas the randomized algorithm along with the VNA-I achieve the worst behaviour, more than 20% worse than the optimal solution.

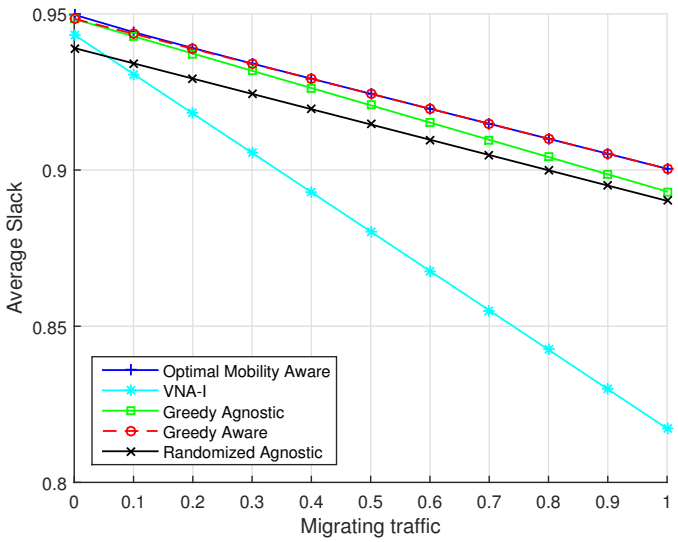

Fig. 6: Average slack

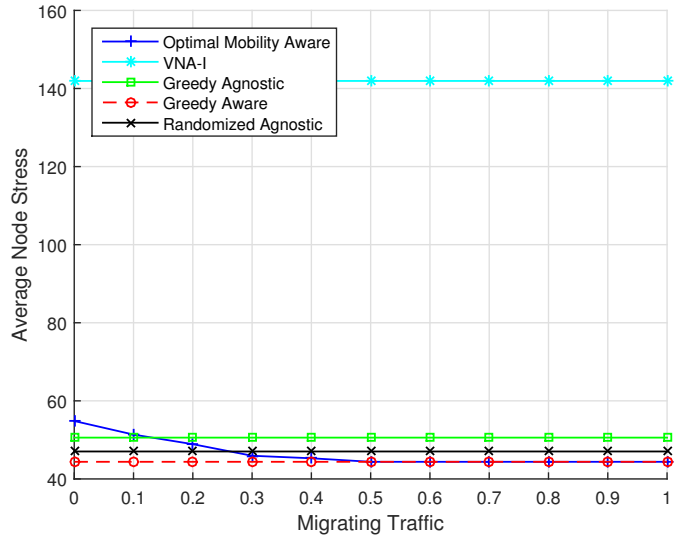

Fig. 7: Average node stress

4.1.2 *Available node capacity slack and node stress*

An important metric for network operators that liaise their physical network infrastructure is the remaining available capacity after the virtual network embedding has been performed; this relates to the additional traffic that the network can accommodate. To this end, Fig. 6 depicts the average node capacity slack in the network. The mobility aware algorithms achieve the biggest average available slack and the rest of the algorithms achieve almost the same performance expect the VNA-I where the average node capacity decreases linearly with the mobility percentage. The same behaviour is observed also for the average node stress as shown in Fig. 7.

4.2 Centralized Mobility Management scheme

After evaluating the performance of the presented algorithms we are interested in studying the effect of the mobility management scheme. As already explained, in the above system model we consider a distributed mobility management scheme. Hence, the mobility function is placed at the very edge of the network on every edge router.

Without altering the algorithms' mapping strategy, we now consider the effect of a centralised mobility management scheme. This means that the mobility function, i.e., mobility anchoring, is placed hierarchically higher in the network. In this case and in contrast to the distributed mobility management scheme, less mobility anchor points serve the scope network domain. Every flow has to be directed through a mobility anchor point and during a handover it will be anchored at this point.

Without loss of generality for the centralized mobility management we consider that a Proxy Mobile IP (PMIPv6) solution is used. PMIPv6 is a network-based mobility solution, where in contrast to the previous mobility management schemes there is no need to install any-mobility related software on the mobile equipment, while the mobility management is assigned to specific network entities. More precisely, Mobility Access Gateway (MAG) is a network entity that is responsible for tracking mobile's location and creating a tunnel with the other basic network entity, Local Mobility Anchor (LMA). LMA, which is an enhanced version of MIPv4s home agent, is mainly responsible for ensuring the reachability of the MNs address, while it moves within a PMIPv6 domain.

When a MN is located inside a PMIPv6 domain, MAG obtains MNs profile and then it sends a Proxy Binding Update (PBU) to LMA, which after an authentication process sends back a Proxy Binding Acknowledgment (PBA) and sets up a route for the MNs home network prefix using the tunnel with the MAG. After receiving the PBA, MAG is able to emulate the MNs home network can send a Router Advertisement (RA) message to MN in order for it to configure its home prefix accordingly. At this moment, all data will be forwarded via this MAG-LMA tunnel, saving bandwidth from the MN by excluding it from mobility-related signalling (so, the MN is unaware of the mobility support procedure).

For the same topology as in the previous simulations we co-locate 4 mobility anchor points that each of them serves 4 destination edge routers. Every path π_{kp} has to include the mobility anchor point that serves the edge router k and every path r_{kji} is now the path that connects the mobility anchor point with the new edge router j where the mobile equipment has migrated to.

4.2.1 *Routing cost*

In the scenario where CMM is in use, the total cost as shown in Fig. 8 and the worst routing of a single flow as shown in Fig. 9 follow the same trend with the previous scenario where DMM was used. However, a 25% increase of the cost is observed and that is expected since now the centralised mobility management directs the flows through non-optimal paths. Hence, the network is becoming more loaded at these parts of the network and the embedding of virtual networks results in an increased use of resources.

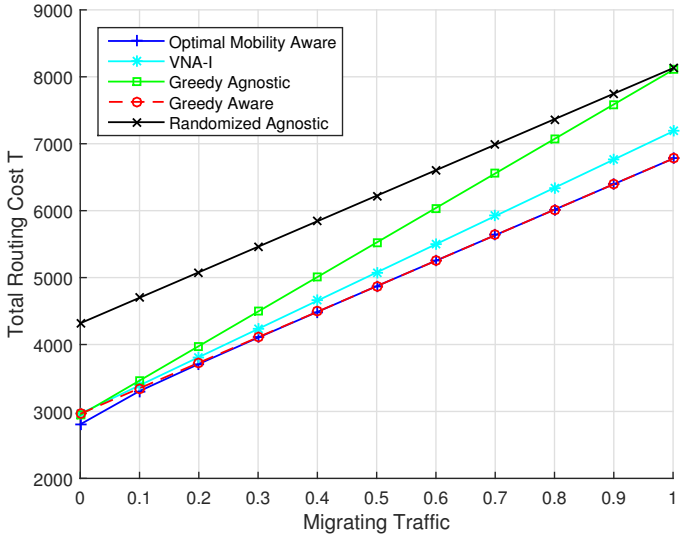

Fig. 8: Total cost as the mobility increases

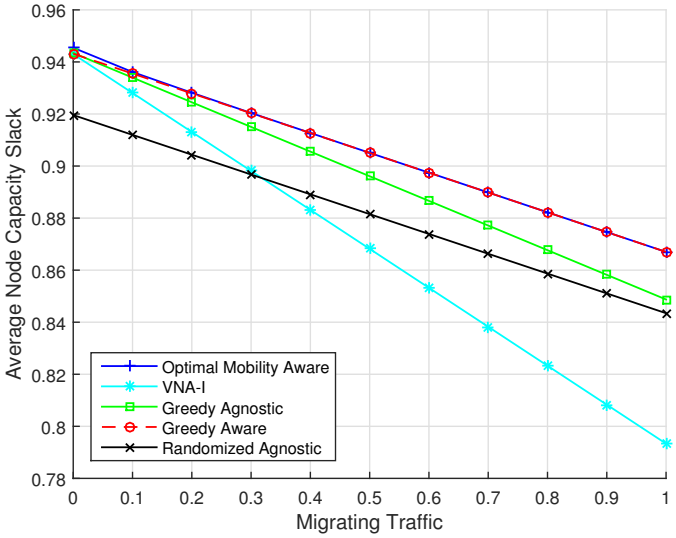

Fig. 10: Average slack

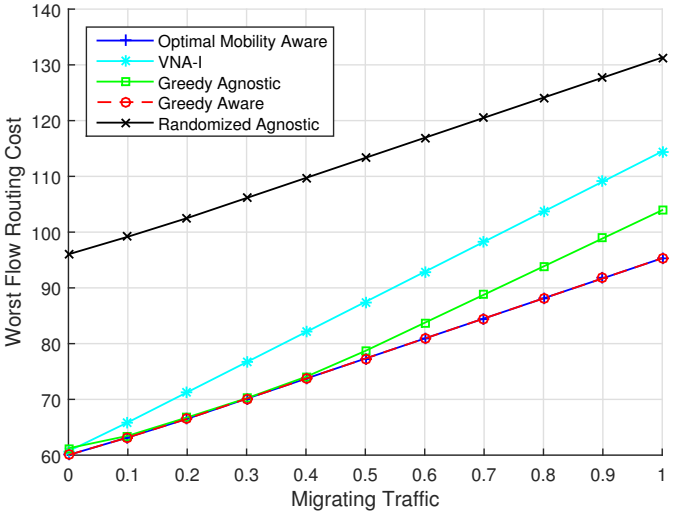

Fig. 9: Worst flow cost as the mobility increases

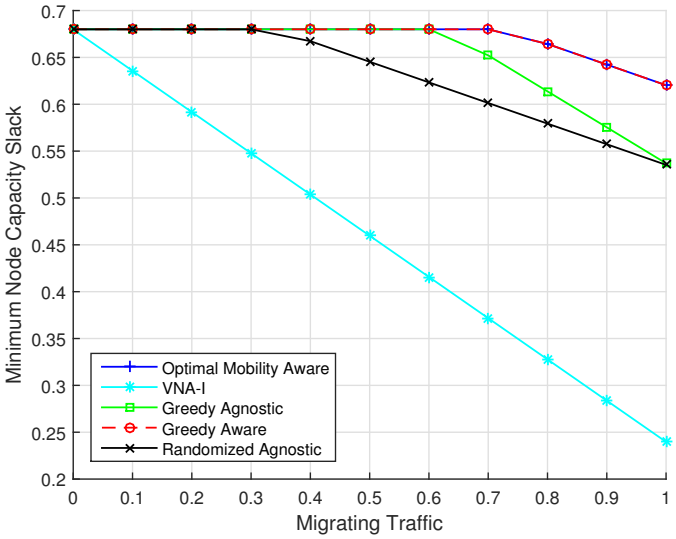

Fig. 11: Minimum node slack

4.2.2 Average node capacity slack and average node stress.

The same behaviour holds for the performance in terms of average and minimum node capacity slack as well as the average node stress where the randomized algorithm and the VNA-I are outperformed by the rest of the algorithms. Again, the mobility aware algorithms achieve the best performance.

4.3 Complexity and optimality gap

As already been eluded above, the performance of the mobility aware optimization framework and the mobility aware greedy heuristic algorithm is the same for a low congestion scenario. The reason is that both of the algorithms load the shortest paths that can hold all the demanded traffic.

However, in terms of complexity, since the optimization algorithm is a Mixed-Integer Linear Programming (MILP), it can be solved in non-polynomial time as it is a well known NP-hard problem. Due to its intractable nature, when the network's topology grows, the proposed optimization algorithm cannot be used. However, from the previously presented results we can see that the proposed greedy mobility aware algorithm can be used with a very competitive performance for efficient virtual network embedding.

However, there is an inherent trade-off between low computational complexity and optimality gap. The embedding problem as it was expressed in this paper could be reformed into a bin-packing problem while the mobility aware greedy algorithm is a form of a next-fit algorithm. According to [30] the optimality gap of the heuristic algorithm can be up to $\times 2$ worst than the optimal solution. For our scenario, the optimality gap depends highly on the selection of the routing paths.

As a last part of the simulation based analysis we are

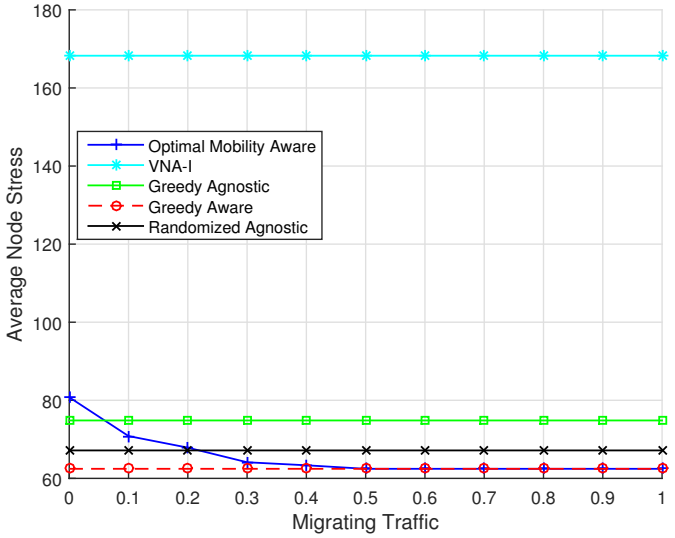

Fig. 12: Average node stress

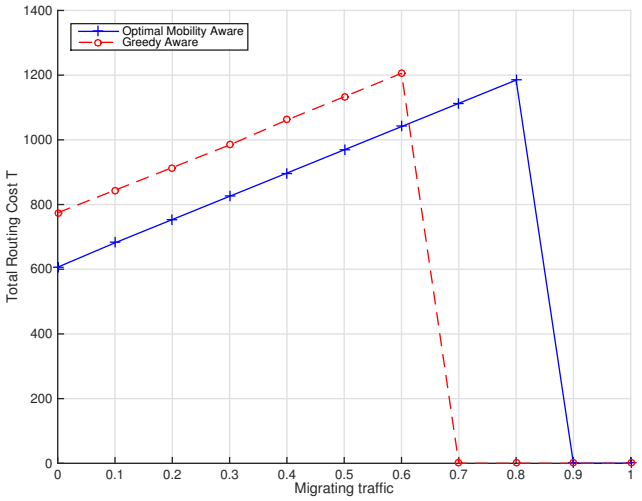

Fig. 13: Optimality gap between optimization and greedy algorithm.

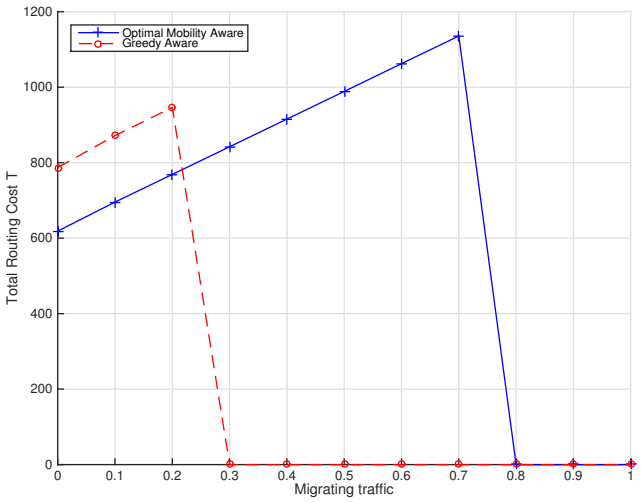

Fig. 14: Optimality gap between optimization and greedy algorithm.

interesting to compare the performance of the two mobility aware algorithms in a high traffic scenario where the topology as well as the available resources may increase the optimality gap between the greedy algorithm and the optimization scheme.

In order to outline that the optimization algorithm potentially outperforms the greedy mobility aware algorithm, we are interested in a high congestion episode scenario where the trivial embedding of the shortest paths in a greedy way does not overlap with the optimal solution.

In this scenario the average node capacity value is set to 10 and the resources' utilization is set to near 1. As Fig. 13 shows, the greedy algorithm has 33% worse performance in terms of total cost T as the migrating traffic due to mobility effect increases. Also, as the congestion increases furthermore due to the migration of the traffic from the mobility effect, the greedy algorithm is no longer able to map the virtual network requests (the zero values mean that no virtual network request was satisfied).

In Fig. 14 we slightly decreased the average node capacity. This resulted in higher network infrastructure utilization and heavier traffic load. In that case it can be observed that the greedy algorithm fails to embed the requested networks at the point of less than half user traffic in comparison to the optimization algorithm.

Depending on the network topology and the network utilization the greedy mobility aware algorithm can potentially be outperformed by the optimization embedding algorithm. However, as it was shown in the previous scenario, when the demand is very low then the greedy strategy maps the virtual networks in the optimal way.

5 Conclusions

With the ever upcoming pressure to increase overall network capacity, while offering almost flat rates to customers for mobile Internet access network, sharing has emerged as a vital component to achieve network sustainability. The expectation is that network sharing via virtualization will move deeper into the core network of operators and will allow for significant lower capital and operational expenditure. A key technical aspect of network virtualization is virtual network embedding, which relates to efficient mapping between physical and virtual resources. The problem of virtual network embedding has been previously studied mainly for fixed networks without taking into account the effect of mobility management solution, which affect routing decisions in case of user mobility.

In this paper, we reveal the impact of distributed mobility management on virtual network embedding and how we might entail in suboptimal virtual networks if mobility is not considered. To this end, we formulate the problem of virtual network embedding as an integer mathematical program in a way that takes explicitly into account the users mobility effect. The simulation results show that the users mobility effect has a significant impact on the virtual network embedding procedure. Numerical investigations

reveal that the performance and efficiency of the network can be significantly increased when the proposed mobility aware algorithm is compared to embedding algorithms that are mobility agnostic.

ACKNOWLEDGMENT

This research is supported by the CROSSFIRE (unCooRdinated netwOrk StrategieS for enhanced interFerence, mobIlity, radio Resource, and Energy saving management in LTE-Advanced networks) project (FP7-PEOPLE-317126) [31] under the "Initial Training Networks" Marie Curie Action. The authors would also like to acknowledge the contribution of their colleagues within the H2020-ICT-2014-2 5G NORMA project; although the views expressed are those of the authors and do not necessarily represent the project.

REFERENCES

[1] M. Nicosia, R. Klemann, K. Griffin, S. Taylor, B. Demuth, J. Defour, R. Medcalf, T. Renger, and P. Datta, "Rethinking Flat Rate Pricing for Broadband Services," *CISCO Internet Business Solutions Group (IBSG)*, Jul. 2012, White Paper.

[2] Chowdhury, N.M.M.K. and Boutaba, R., "Network Virtualization: State of the Art and Research Challenges," *Communications Magazine, IEEE*, vol. 47, no. 7, pp. 20–26, July 2009.

[3] N. M. K. Chowdhury and R. Boutaba, "A Survey of Network Virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862 – 876, 2010.

[4] A. Fischer, J. Botero, M. Till Beck, H. de Meer, and X. Hesselbach, "Virtual Network Embedding: A Survey," *Communications Surveys Tutorials, IEEE*, vol. 15, no. 4, pp. 1888–1906, Apr. 2013.

[5] C. Perkins, "IP Mobility Support for IPv4, Revised," RFC 5944 (Proposed Standard), IETF, Nov. 2010.

[6] H. Chan, D. Liu, P. Seite, H. Yokota, and J. Korhonen, "Requirements for Distributed Mobility Management," Internet Draft: draft-ietf-dmm-requirements-15 (work-in-progress), IETF, Mar. 2014.

[7] D. Liu, J. Zuniga, P. Seite, H. Chan, and C. Bernandos, "Distributed Mobility Management: Current practices and gap analysis ," Internet Draft: draft-ietf-dmm-best-practices-gap-analysis-03, IETF, Aug. 2014.

[8] G. Chochlidakis and V. Friderikos, "Mobility Aware Virtual Network Embedding," in *IEEE International Conference on Communications (ICC)*, London, United Kingdom, Jun. 2015.

[9] "Open Networking Foundation, White paper: Software Defined Networking: The New Norm for Networks," Apr. 2012.

[10] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[11] R. Enns and M. Bjorklund and J. Schonwalder and A. Bierman, "NETCONF Configuration Protocol," RFC 6241 (Proposed Standard), IETF, Jun. 2011.

[12] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Technical Report: FlowVisor: A Network Virtualization Layer," no. OPENFLOW-TR-2009-2, Oct. 2009.

[13] ——, "Can the Production Network Be the Testbed?" in *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 1–6.

[14] S. Gutz, A. Story, and N. Foster, "Splendid Isolation: A Slice Abstraction for Software-Defined networks," in *in ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, 2012.

[15] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in sdn-openflow networks," *Computer Networks*, vol. 71, no. 0, pp. 1 – 30, 2014.

[16] N. Operators, "Network Functions Virtualization, An Introduction, Benefits, Enablers, Challenges and Call for Action," in *SDN and OpenFlow SDN and OpenFlow World Congress*, 2012.

[17] R. Bolla, C. Lombardo, R. Bruschi, and S. Mangialardi, "DROPv2: Energy Efficiency Through Network Function Virtualization," *Network, IEEE*, vol. 28, no. 2, pp. 26–32, March 2014.

[18] A. Basta, W. Kellerer, M. Hoffmann, H. J. Morper, and K. Hoffmann, "Applying NFV and SDN to LTE Mobile Core Gateways, the Functions Placement Problem," in *Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications, & Challenges*, ser. AllThingsCellular '14. New York, NY, USA: ACM, 2014, pp. 33–38.

[19] V. Yazici, U. Kozat, and M. Sunay, "A New Control Plane for 5G Network Architecture with a Case Study on Unified Handoff, Mobility, and Routing Management," *Communications Magazine, IEEE*, vol. 52, no. 11, pp. 76–85, Nov 2014.

[20] N. Chowdhury, M. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," in *INFOCOM 2009, IEEE*, Apr. 2009, pp. 783–791.

[21] M. Melo, S. Sargento, U. Killat, A. Timm-Giel, and J. Carapinha, "Optimal Virtual Network Embedding: Node-Link Formulation," *Network and Service Management, IEEE Transactions on*, vol. 10, no. 4, pp. 356–368, Dec. 2013.

[22] G. Hernando, S. Perez, and J. Cabero, "Mobility-Aware Distributed Embedding (MADE) of Virtual Networks," in *Future Network and Mobile Summit, 2010*, Jun. 2010, pp. 1–8.

[23] I. Houidi, W. Louati, and D. Zeghlache, "A Distributed Virtual Network Mapping Algorithm," in *Communications, 2008. ICC '08. IEEE International Conference on*, May 2008, pp. 5634–5640.

[24] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "VNE-AC: Virtual Network Embedding Algorithm Based on Ant Colony Metaheuristic," in *Communications (ICC), 2011 IEEE International Conference on*, Jun. 2011, pp. 1–6.

[25] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 17–29, Mar. 2008.

[26] Z. Cai, F. Liu, N. Xiao, Q. Liu, and Z. Wang, "Virtual Network Embedding for Evolving Networks," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, Dec. 2010, pp. 1–5.

[27] R. Gomes, L. Bittencourt, and E. Madeira, "A Bandwidth-Feasibility Algorithm for Reliable Virtual Network Allocation," in *Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on*, May 2014, pp. 504–511.

[28] Y. Zhu and M. Ammar, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, April 2006, pp. 1–12.

[29] C. H. Papadimitriou, "On the Complexity of Integer Programming," *J. ACM*, vol. 28, no. 4, pp. 765–768, Oct. 1981.

[30] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York, NY, USA: John Wiley & Sons, Inc., 1990.

[31] The CROSSFIRE project (FP7- Initial Training Network Marie Curie). [Online]. Available: http://gain.di.uoa.gr/crossfire/

Giorgos Chochlidakis graduated from the school of Electrical and Computer Engineering of the National Technical University of Athens (NTUA). Since 2013 is a member of the Centre for Telecommunications Research (CTR) of the departments of Informatics of King's College London (KCL) as a PhD candidate under the supervision of Dr Vasilis Friderikos. He, also, participates to the European Initial Training Networks (ITN) Marie Curie action project named CROSSFIRE (unCooRdinated netwOrk StrategieS for enhanced interFerence, mobIlity, radio Resource, and Energy saving management in LTE-Advanced networks). His research interests focus on network virtualization, mobility management for next-generation networks, virtual network embedding and network sharing.

Vasilis Friderikos published 200 research papers in flagship IEEE, Elsevier, Springer journals, international conferences, book chapters and patents. He has been program co-chair of IEEE ICT'16 and co-chair at the IEEE WCNC 2010 conference (acting technical program committee member for IEEE Globecom, IEEE ICC and several other flagship international conferences). He has also been organizing committee member of the Green Wireless Communications and Networks Workshop (GreeNet) during VTC Spring 2011. He has been teaching advanced mobility management protocols for the Future Internet at the Institut Suprieur de l'Electronique et du Numrique (ISEN) in France during autumn 2010. Received two times best paper awards in IEEE ICC 2010 and WWRF conferences respectively. He has been visiting researcher at WinLab in Rutgers University (USA) and recipient of the British Telecom Fellowship Award in 2005. Vasilis is a member of IEEE, member of IET and member of the INFORMS section on Telecommunications His research interests lie broadly within the closely overlapped areas of wireless networking, mobile computing, and architectural aspects of the Future Internet. The emphasis is on the design and analysis of algorithms for scheduling, routing, admission control, load and power management in virtualized wireless networks with application to both centralized and distributed implementations.